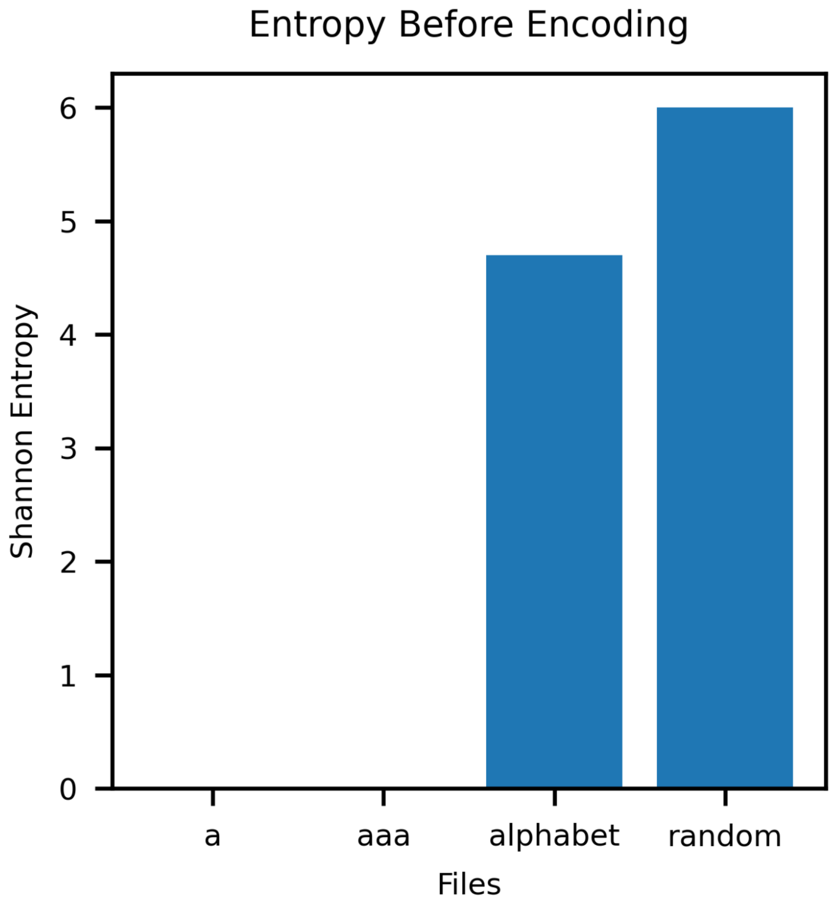
Nicholas Chan
nipchan@ucsc.edu
5/09/2021
CSE13s Spring ‘21

Writeup:
Assignment 5: Hamming Codes

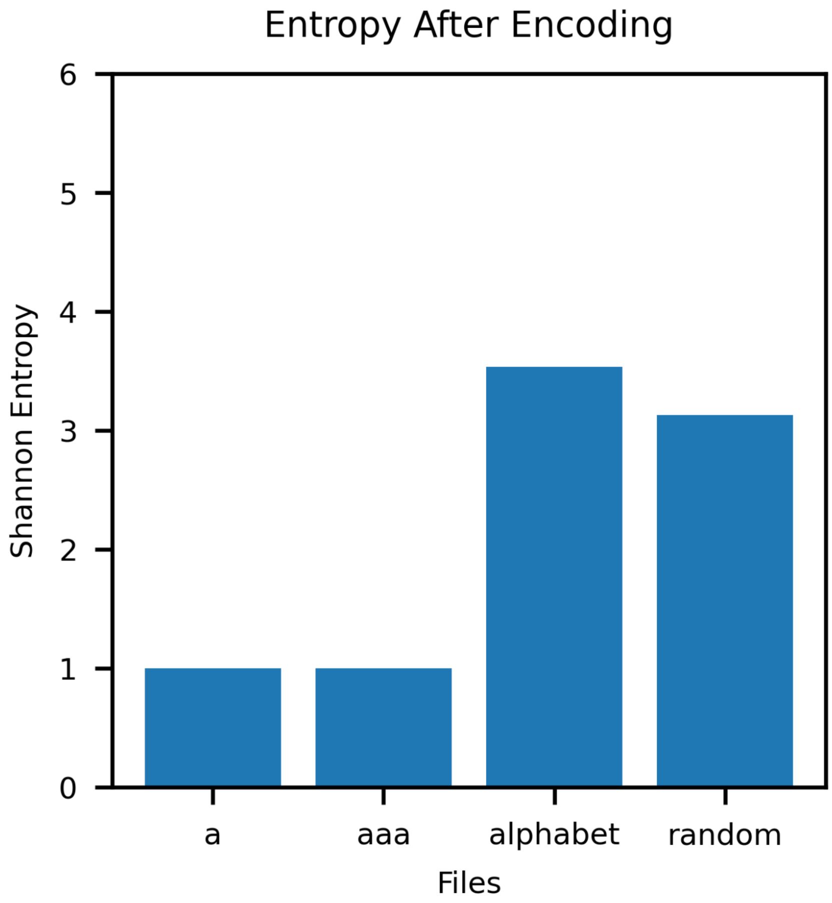
For assignment 5, I implemented an encoder and decoder for Hamming codes. The scheme that this assignment used was a Hamming (8,4), where the lower nibble of a byte would be reserved for data bits. The upper nibble of the byte would holde parity bits for error detection and correction. Statistics on bytes processed, errors corrected, uncorrectable errors and error rates are available by enabling the verbose argument on the decode module.

The bar graph below displays the entropies observed among a sample of text files under the artificial directory in corpora from the cse13s-resources repo.



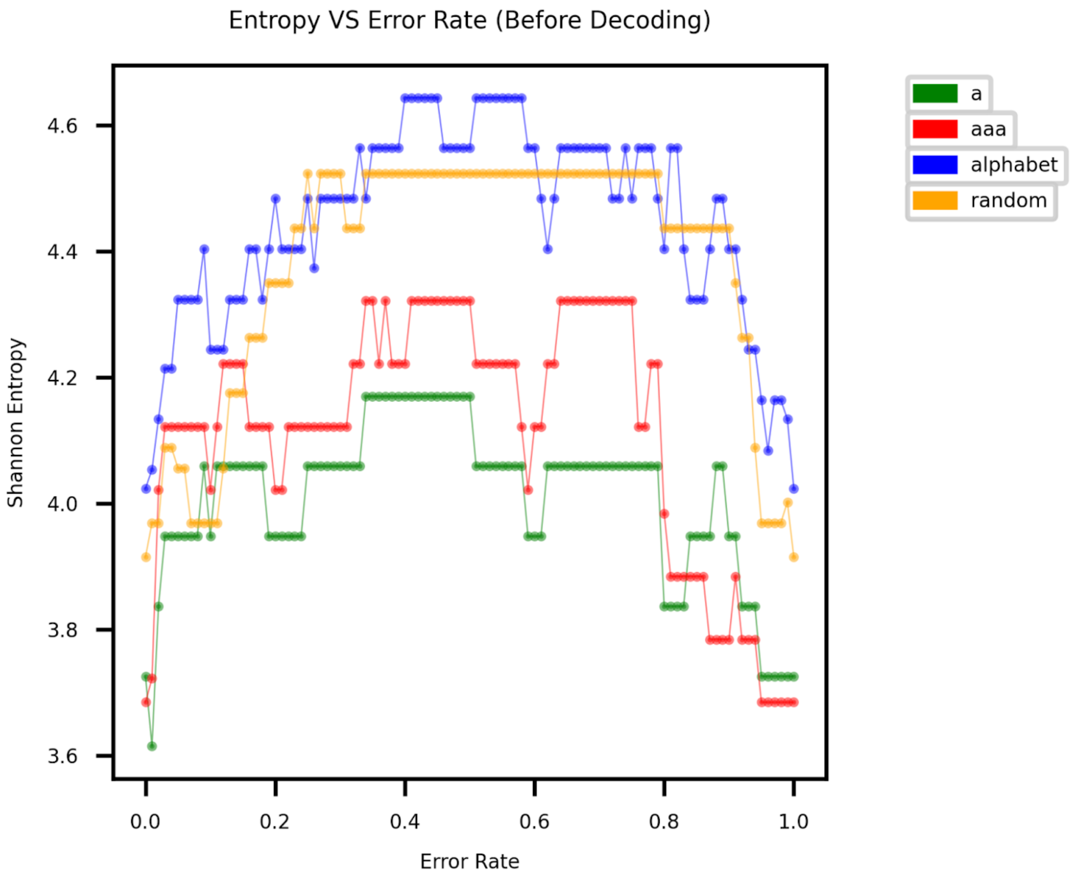
The files a.txt and aaa.txt vary extremely in size, with aaa holding an enormous count of characters, however both only exhibit 1 variety of character, ‘a’. On the other hand, the files alphabet.txt and random.txt exhibit a large number of characters as well as a wide variety. As expected, the alphabet is lower than random as only alphabet characters are included. The file alphabet.txt can thus be classified as a true subset of the random.txt file which holds all types of characters including alphabetical characters. This histogram displays the effect of variety and not just quantity on Shannon entropy, which is essentially a measure of chaos.

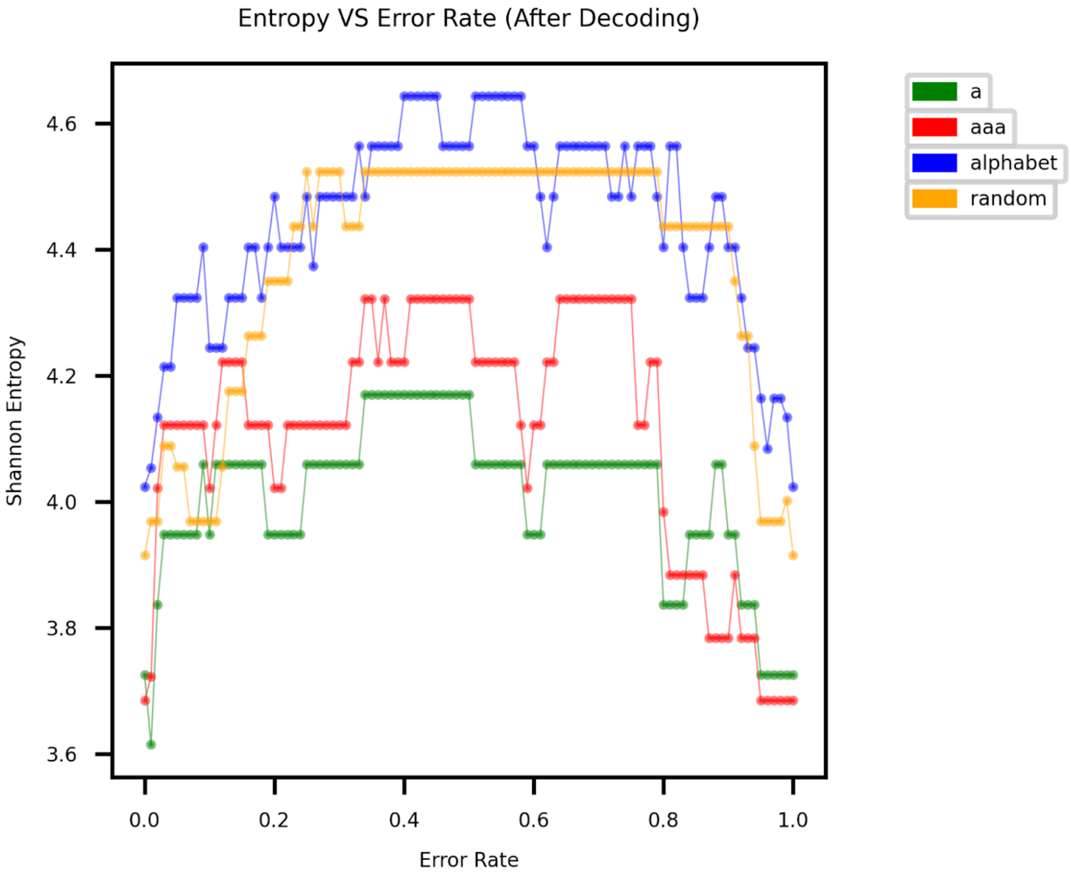
The next bar graph below displays the entropies observed among a sample of text files under the artificial directory in corpora from the cse13s-resources repo after encoding



Due to the nature of encoding, the characters read from this file won't make sense as they are bytes of hamming code. However the effect of the encoding process on the variety of characters exhibited becomes clear with this graph. The fact that the bars for a.txt and aaa.txt are now visible denote that some increase in variety was caused among their respective characters. However, the encoding process isn't random. The only explanation to this would likely be hidden characters like the return line character which would have been transformed along with the character 'a'. Taking a look at the entropies of alphabet.txt and random.txt we notice that alphabet is more chaotic than random. From what we said before about how a and aaa were altered, it is pretty hard to see what is going on. However if we take a step back, we notice that the range of entropy for the encoded is lower overall than it was before encoding. This observation helps us make more sense out of the varying changes observed among the encoded a, aaa, alphabet and random .txt files as the encoding process seems to somehow normalize all of their measures of entropies. Considering the matrix multiplication occurring in the encoding process, this does make sense as the same generator matrix is used to convert each of the characters in these files. Because of this, files with more character diversity may see some more similarities as the parity bits added in the encoding process reduce the total chaos observed.

Below are plots of the Shannon entropies observed among a, aaa, alphabet and random .txt files over varying error rates before and after the decoding process.





We will notice that there appears to be no difference observed between the two graphs in terms of the range and distribution of shannon entropies over varying error rates.

Under the same conditions for the introduction of noise, the fact that the entropies observed before and after encoding are the same exhibit some property on monotonic behavior between Shannon entropy, error rate and decoding. However, something else to notice is where the distributions of Shannon entropy over error rate lie for a, aaa, alphabet and random. The trend exhibited by all of these graphs resemble the normal distribution, with total entropy falling off as the error rate minimizes and maximizes. The conditions of minimum and maximum error rates seem to apply a form of consistency, with the decrease in the growth rates of entropy around those regions being evidence. The most centered values for error rate seem to minimize the consistency applied characters in a file (the optimal error rate for screwing up messages and increasing entropy seems to be around 0.3-6).

Other observations about these distributions that make sense are how the entropies among a, aaa, alphabet and random are consistent with how the were before and after encoding, with a and aaa exhibiting the lowest entropies and alphabet and random exhibiting the highest entropies.

Although there hasn't been a significant change observed in the entropies observed after decoding, that shouldn't be a source of alarm as that denotes that our decoded messages shouldn't have strayed too far off from the transformations we applied to each of them in the form of matrix multiplication with the constant matrices G and Ht (The generator matrix and transpose of the parity checker matrix).